M-Bus to LoRaWAN - Integration manual

An integration manual for the ACRIOS Systems's converters ACR-CV-101L-M-D and ACR-CV-101L-M-EAC. The converter is being used to retrofit existing meters with M-Bus communication to a LoRaWAN communication.

Introduction

The documentation needed for the ACRIOS Systems M-Bus to LoRaWAN concentrator. The documentation refers to the default LUA script used and will cover possible configuration options.

Typical use-case

In the realm of district heating stations, efficient monitoring and management of heat meters play a crucial role in optimizing energy consumption and enhancing overall operational effectiveness. The integration of the M-Bus to LoRaWAN converter presents a streamlined solution that facilitates rapid deployment and seamless connectivity for heat meter data, ensuring real-time insights and efficient control.

Scenario:

Imagine a district heating station serving a residential complex. This facility relies on a network of heat meters installed within individual apartments to monitor heat consumption accurately. The challenge is to efficiently collect and transmit this consumption data to a central management system for billing, monitoring, and maintenance purposes.

Solution:

The M-Bus to LoRaWAN converter provides a plug-and-play solution to address this challenge.

Benefits:

- **Time and Cost Savings:** The quick installation process eliminates extensive configuration and wiring efforts, reducing installation time and associated costs.
- **Seamless Integration:** The plug-and-play nature of the solution simplifies the integration of heat meters into the district heating network, minimizing disruption to residents.
- **Reliable Communication:** The external antenna ensures reliable signal propagation, maintaining consistent communication even in complex architectural environments.
- **Real-time Insights:** The solution provides real-time heat consumption data, enabling accurate billing and informed decision-making for energy optimization.

What is M-Bus?

M-Bus (Meter-Bus) is a European standard (EN 13757-2 physical and link layer, EN 13757-3 application layer) for the remote reading of water meter, gas or electricity meters. M-Bus is also usable for other types of consumption meters. The M-Bus interface is made for communication on two wires, making it cost-effective. A radio variant of M-Bus (Wireless M-Bus) is also specified in EN 13757-4.

Data on the M-Bus is transferred in telegrams which consist of one or more frames, there must be no pauses within telegrams, not even after stop bits. M-Bus uses four different telegram formats with following structures:

Single Character	Short Frame	Control Frame	Long Frame
E5h	Start 10h	Start 68h	Start 68h
	C Field	L Field = 3	L Field
	A Field	L Field = 3	L Field
	Check Sum	Start 68h	Start 68h
	Stop 16h	C Field	C Field
		A Field	A Field
		CI Field	CI Field
		Check Sum	User Data (0-252 Byte)
		Stop 16h	Check Sum
			Stop 16h

> ! When communicating with more meters on the same M-Bus line, the addressing is crucial (A Field). Meter can be addressed by its primary address or by its secondary address.

Primary addressing:

Address	**Function**
0	Factory default address.
1-250	Addresses that can be assigned to slaves.(Primary addresses)
251-252	Reserved for future use.
253	Indicates that addressing is performed at network layer instead. (secondary addressing procedure)
254	Broadcast, meters reply with their addresses.(Causes collision, test purposes only.)
255	Broadcast, meters do not reply.

The secondary address consists of 4 parts:

- 4 bytes being the device ID (serial #)
- 2 bytes being the manufacturer's identifier
- 1 byte being the device version
- 1 byte being the device media

Benefit of secondary addressing is that you do not have to reconfigure meter's primary addresses and can navigate the installation only by modifying the converters' addressing mechanism.

Converter integration

Functions

- Configurable payload length, by default 48B
- Configurable acknowledged messages with 3 retransmission. If no ACK ⇒ retransmission.
- Configurable reading period - by default 2h 30min
- Configurable initial M-bus delay - by default 2 000 ms
- Configurable M-Bus parameters
- Configurable port
- Configurable VIF DIF filtering

Out of the box behavior

By default after connecting the meter to the converter, the device will send a broadcast query on address 254 and the received data are being forwarded to the LoRaWAN network in its original format.

> 💡 If the meter is not responding, the payload “NO DATA RECEIVED” in its HEX format 4E 4F 20 44 41 54 41 20 52 45 43 45 49 56 45 44 will be transmitted.

To avoid sending longer payloads then allowed by the network depending on the current SF, by default the converter splits the received data into the 51B blocks where the first byte identifies the specific payload, second byte identifies the total number of payloads and the data follows.

For example if the whole payload is 240 bytes long, 5 separate LoRaWAN messages are being sent:

01 05 XX XX XX…

02 05 XX XX XX…

03 05 XX XX XX…

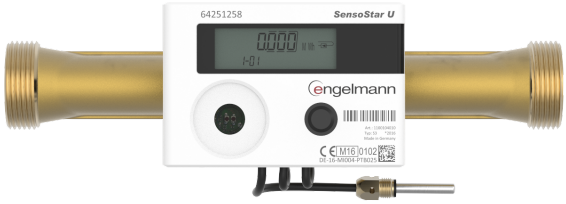
04 05 XX XX XX…

05 05 XX XX XX…

Engelmann SensoStar U read out

From M-Bus meter

```
00 : 68 B3 B3 68 08 00 72 98 53 85 30 C5 14 00 0A 8D
10 : 10 00 00 04 78 E6 D0 D6 01 04 06 00 00 00 00 04
20 : 13 00 00 00 00 04 2B 00 00 00 00 14 2B 00 00 00
30 : 00 04 3B 00 00 00 00 14 3B 00 00 00 00 02 5B 18
40 : 00 02 5F 18 00 02 61 C9 FF 02 23 B9 00 04 6D 08
50 : 2A FC 28 44 06 00 00 00 00 44 13 00 00 00 00 42
60 : 6C 00 00 01 FD 17 10 03 FD 0C 05 00 00 84 20 06
70 : 00 00 00 00 C4 20 06 00 00 00 00 84 30 06 00 00
80 : 00 00 C4 30 06 00 00 00 00 84 40 13 00 00 00 00
90 : C4 40 13 00 00 00 00 84 80 40 13 00 00 00 00 C4
A0 : 80 40 13 00 00 00 00 84 C0 40 13 00 00 00 00 C4
B0 : C0 40 13 00 00 00 00 95 16
```



Engelmann SensoStar U with M-Bus module

Packet 1 out of 4

```
00 : 01 04 68 B3 B3 68 08 00 72 98 53 85 30 C5 14 00
10 : 0A 8D 10 00 00 04 78 E6 D0 D6 01 04 06 00 00 00
20 : 00 04 13 00 00 00 00 04 2B 00 00 00 00 14 2B 00
30 : 00 00
```

Sending packet 2 out of 4

```
00 : 02 04 00 04 3B 00 00 00 00 14 3B 00 00 00 00 02
10 : 5B 18 00 02 5F 18 00 02 61 C9 FF 02 23 B9 00 04
20 : 6D 08 2A FC 28 44 06 00 00 00 00 44 13 00 00 00
30 : 00 42
```

Sending packet 3 out of 4

```
00 : 03 04 6C 00 00 01 FD 17 10 03 FD 0C 05 00 00 84
10 : 20 06 00 00 00 00 C4 20 06 00 00 00 00 84 30 06
20 : 00 00 00 00 C4 30 06 00 00 00 00 84 40 13 00 00
30 : 00 00
```

Sending packet 4 out of 4

```
00 : 04 04 C4 40 13 00 00 00 00 84 80 40 13 00 00 00
10 : 00 C4 80 40 13 00 00 00 00 84 C0 40 13 00 00 00
20 : 00 C4 C0 40 13 00 00 00 00 95 16
```

Payload parsing

The payload itself is in a form of typical M-Bus frame, including the header and any M-Bus parser can be used. We recommend using one of the parsers below. Please always check for a possible licensing model.

💡 **Alexander Miller M-Bus parser**
https://www.miller-alex.de/Mbus

💡 **jMBus parser**
https://www.openmuc.org/m-bus/

💡 **libmbus - M-bus Library from Raditex Control**
https://github.com/rscada/libmbus

💡 **ACRIOS Backend**
https://backend.wmbus.acrios.com/docs#/Parsers/parse_provided_input_as_hex_mbus_parser_mbus_hex_get

Hardware used

1. ACR-CV-101L-M-D - M-Bus to LoRaWAN converter, battery powered
2. ACR-CV-101L-M-EAC - M-Bus to LoRaWAN converter, externally powered

The converter allows you to connect any meter or other device equipped with an M-Bus communication.

Hardware limitations

- Hardware allows you to connect up to 5UL. 1UL equals to 1.5mA. Please note that 1UL is not always a representation of 1 meter as some meters require 2UL and sometimes more. This information should be stated on the documentation of the M-Bus meter itself.
- It is important not to connect 2 M-Bus masters at once.
- The typical battery life time is:

Reading and sending interval	Battery life time
15 min	1,7 years
30 min	3,9 years
60 min	5,2 years

> ! The **longer initial delay** on the M-Bus line translates to **higher power consumption**. In case of externally powered version it may not be a problem, however with a battery operated units it might be a constraint.

Application limitations

- From the nature of M-Bus communication, when more than 1 device is connected it is needed to address those meters separately. Either via primary or secondary addressing.
- From the nature of LoRaWAN communication it is needed to shorten the payload. As the M-Bus frame can be up to 240 bytes long and you can transfer only up to 222 bytes while using SF7 or SF8 it might be needed to either shorten the payload by filtering the VIF DIF values on the hardware level or by splitting the telegram into multiple smaller telegrams.

LUA script configuration

The LUA scripting interface plays a pivotal role in the functionality of our M-Bus to LoRaWAN converter, enabling advanced customization and protocol integration. LUA scripting, a lightweight scripting language, empowers us to enhance and adapt the converter's capabilities by creating and executing scripts.

The default script for our M-Bus to LoRaWAN converter can be found here:

https://sw.acrios.com/acrios/acr-cv-lua/-/blob/a27325fd4720d5be159aab86ea3b6f35b7116a21/ACR_CV_101L_M_X_lorawanMbus.lua

Main parts of the script

Configuration

The main part used to configure a device wake-up interval or basic M-Bus parameters. You can also configure the acknowledged message mechanism, used port or maximum payload size.

```
---- CONFIGURATION ----
------- LoRaWAN -------
ack = 0                -- 1 for acknowledged, 0 for non-acknowledged
port = 100             -- transmit port
receiveTimeout = 10000 -- the maximum execution time in milliseconds
payload_size = 48      -- payload size (total message size is 48B payload + 2B header). Maximum payload size with SF12 / DR
0 is 51B
------- M-BUS ---------
baudrate = 2400   -- baudrate: up to 921600 baud
parity = 2        -- communication parity: 0 for none, 1 for odd and 2 for even parity
stopBits = 1    -- number of stop bits: 1 or 2
dataBits = 8    -- number of data bits: 7 or 8
initialDelay = 2000     -- delay before sending request - some devices require 3000 (Engelmann SensoStar), impacts battery
 life!
------ Timing ---------
-- device wakeup interval
periodHours = 2
periodMinutes = 30
-- CONFIGURATION END --
```

Primary addressing of one meter

The code on the line #61 is being used as a configuration for the meter readout. By default the device is addressing the address 254 to which the connected meter should answer.

> ! If the device is not answering, you might want to check the "*initialDelay*" parameter under the configuration and set it to a higher value. The initialDelay is a time for which the M-Bus is being powered on but with no activity. Some meters required this time to be 2 seconds but some devices might need up to 6 seconds.
>
> From our experience an Engelmann devices needs 3 seconds and the Schneider Electric devices iEM needs **up to 6 seconds**.

Example of M-Bus query - address 254

```
  b=pack.pack('<b5', 0x10, 0x7B, 0xFE, (0x7B+0xFE)%256, 0x16)

  status,_,_,_,_,raw = api.mbusTransaction(b,3000,1)

-- CREATE MBUS QUERY --
    -- MBUS Query is being assembled as described in MBUS protocol documentation. In the example below broadcast adddress 2
54 (=0xFE) is used.
    -- Documentation https://m-bus.com/documentation-wired/05-data-link-layer
    -- Request for Class 2 Data - REQ_UD2
    -- b=pack.pack('<b5', 0x10, 0x5B, 0xFE, 0x59, 0x16)
    -- 0x10 - Start byte
    -- 0x5B or 0x7B - C-Field - Request for Class 2 Data
    -- 0xFE - Address field
    -- 0x59 - CRC - calculated by (0x7B+0xFE)%256
    -- 0x16 - Stop byte
```

💡 Address 254 is used as a broadcast address to which any meter answers. It is good to use it in cases that requires reading only from one connected meter without having to deal with primary addressing.

Example of M-Bus query - address 1

```
  b=pack.pack('<b5', 0x10, 0x7B, 0x01, (0x7B+0x01)%256, 0x16)

  status,_,_,_,_,raw = api.mbusTransaction(b,3000,1)

-- CREATE MBUS QUERY --
    -- MBUS Query is being assembled as described in MBUS protocol documentation. In the example below broadcast adddress 2
54 (=0xFE) is used.
    -- Documentation https://m-bus.com/documentation-wired/05-data-link-layer
    -- Request for Class 2 Data - REQ_UD2
    -- b=pack.pack('<b5', 0x10, 0x5B, 0xFE, 0x59, 0x16)
    -- 0x10 - Start byte
    -- 0x5B or 0x7B - C-Field - Request for Class 2 Data
    -- 0x01 - Address field
    -- 0x59 - CRC - calculated by (0x7B+0x01)%256
    -- 0x16 - Stop byte
```

Primary addressing of multiple meters

💡 Hardware allows you to connect up to 5UL. 1UL equals to 1.5mA. Please note that 1UL is not always a representation of 1 meter as some meters might require 2UL and in some cases even more. This information should be stated on the documentation of the M-Bus meter itself.

Example of M-Bus query - primary address 1 & 2

```
-- CREATE MBUS QUERY --

    b1=pack.pack('<b5', 0x10, 0x7B, 0x01, (0x7B+0x01)%256, 0x16)
    b2=pack.pack('<b5', 0x10, 0x7B, 0x02, (0x7B+0x02)%256, 0x16)    --- New line (this one) has to be added

    status,_,_,_,_,raw1 = api.mbusTransaction(b1,3000,1)
    status,_,_,_,_,raw2 = api.mbusTransaction(b2,3000,1)            --- New line (this one) has to be added

    raw = raw1 .. raw2
```

Read out of 2 meters & both answers

Read out of the 2 connected devices on addresses 0x00 and 0x0B. The data are stick together and it is possible to filter them by searching for a valid M-Bus frame starting with “68” and ending with “16”.

From MBus Meter:

```
00 : 68 56 56 68 08 0B 72 14 05 00 15 77 04 14 03 37
10 : 30 00 00 0C 78 14 05 00 15 0D 7C 08 44 49 20 2E
20 : 74 73 75 63 0A 37 30 34 33 30 30 42 42 35 31 04
30 : 6D 27 0A FD 28 02 7C 09 65 6D 69 74 20 2E 74 61
40 : 62 F2 08 04 13 CE 00 00 00 04 93 7F 00 00 00 00
50 : 44 13 CE 00 00 00 0F 10 02 1F 87 16 68 B3 B3 68
60 : 08 00 72 98 53 85 30 C5 14 00 0A C5 10 00 00 04
70 : 78 E6 D0 D6 01 04 06 00 00 00 00 04 13 00 00 00
80 : 00 04 2B 00 00 00 00 14 2B 00 00 00 00 04 3B 00
90 : 00 00 00 14 3B 00 00 00 00 02 5B 17 00 02 5F 17
A0 : 00 02 61 0C 00 02 23 BA 00 04 6D 32 28 FD 28 44
B0 : 06 00 00 00 00 44 13 00 00 00 00 42 6C 00 00 01
C0 : FD 17 10 03 FD 0C 05 00 00 84 20 06 00 00 00 00
D0 : C4 20 06 00 00 00 00 84 30 06 00 00 00 00 C4 30
E0 : 06 00 00 00 00 84 40 13 00 00 00 00 C4 40 13 00
F0 : 00 00 00 84 80 40 13 00 00 00 00 C4 80 40 13 00
00 : 00 00 00 84 C0 40 13 00 00 00 00 C4 C0 40 13 00
10 : 00 00 00 39 16
```

Engelmann SensoStar U & Itron M-Bus Cyble v2.0

Sending packet 1 out of 6

```
00 : 01 06 68 56 56 68 08 0B 72 14 05 00 15 77 04 14
10 : 03 37 30 00 00 0C 78 14 05 00 15 0D 7C 08 44 49
20 : 20 2E 74 73 75 63 0A 37 30 34 33 30 30 42 42 35
30 : 31 04
```

Sending packet 2 out of 6

```
00 : 02 06 6D 27 0A FD 28 02 7C 09 65 6D 69 74 20 2E
10 : 74 61 62 F2 08 04 13 CE 00 00 00 04 93 7F 00 00
20 : 00 00 44 13 CE 00 00 00 0F 10 02 1F 87 16 68 B3
30 : B3 68
```

Sending packet 3 out of 6

```
00 : 03 06 08 00 72 98 53 85 30 C5 14 00 0A C5 10 00
10 : 00 04 78 E6 D0 D6 01 04 06 00 00 00 00 04 13 00
20 : 00 00 00 04 2B 00 00 00 00 14 2B 00 00 00 00 04
30 : 3B 00
```

Sending packet 4 out of 6

```
00 : 04 06 00 00 00 14 3B 00 00 00 00 02 5B 17 00 02
10 : 5F 17 00 02 61 0C 00 02 23 BA 00 04 6D 32 28 FD
20 : 28 44 06 00 00 00 00 44 13 00 00 00 00 42 6C 00
30 : 00 01
```

Sending packet 5 out of 6

```
00 : 05 06 FD 17 10 03 FD 0C 05 00 00 84 20 06 00 00
10 : 00 00 C4 20 06 00 00 00 00 84 30 06 00 00 00 00
20 : C4 30 06 00 00 00 00 84 40 13 00 00 00 00 C4 40
30 : 13 00
```

Sending packet 6 out of 6

```
00 : 06 06 00 00 00 84 80 40 13 00 00 00 00 C4 80 40
10 : 13 00 00 00 00 84 C0 40 13 00 00 00 00 C4 C0 40
20 : 13 00 00 00 00 39 16
```

Read out of 2 meters & only one answers

Read out of the 2 connected devices on addresses 0x00 and 0x0B. Only the data from communicating device are forwarded. It is possible to identify the device by its header within the M-Bus protocol

From MBus Meter:

```
00 : 68 56 56 68 08 0B 72 14 05 00 15 77 04 14 03 38
10 : 30 00 00 0C 78 14 05 00 15 0D 7C 08 44 49 20 2E
20 : 74 73 75 63 0A 37 30 34 33 30 30 42 42 35 31 04
30 : 6D 30 0A FD 28 02 7C 09 65 6D 69 74 20 2E 74 61
40 : 62 F2 08 04 13 CE 00 00 00 04 93 7F 00 00 00 00
50 : 44 13 CE 00 00 00 0F 10 02 1F 91 16
```

Itron M-Bus Cyble v2.0

Sending packet 1 out of 2

```
00 : 01 02 68 56 56 68 08 0B 72 14 05 00 15 77 04 14
10 : 03 38 30 00 00 0C 78 14 05 00 15 0D 7C 08 44 49
20 : 20 2E 74 73 75 63 0A 37 30 34 33 30 30 42 42 35
30 : 31 04
```

Sending packet 2 out of 2

```
00 : 02 02 6D 30 0A FD 28 02 7C 09 65 6D 69 74 20 2E
10 : 74 61 62 F2 08 04 13 CE 00 00 00 04 93 7F 00 00
20 : 00 00 44 13 CE 00 00 00 0F 10 02 1F 91 16
```

Read out of 2 meters & none answers

Read out of the 2 connected devices on addresses 0x00 and 0x0B. Only the message “NO DATA RECEIVED” is sent as no meter sends any data.

From MBus Meter:

```
00 : 01 01 4E 4F 20 44 41 54 41 20 52 45 43 45 49 56
10 : 45 44
```

💡 4E 4F 20 44 41 54 41 20 52 45 43 45 49 56 45 44 HEX = NO DATA RECEIVED ASCII

Filtering on the hardware level

To avoid sending multiple messages on the LoRaWAN network with each reading, it is possible to filter data on the hardware level. It is however important to note, that the filter can be applied on the specific meter type / meter manufacturer.

To do so, the device can shorten the payload by giving it the required VIF and DIF values of the data that needs to be forwarded to the network server. The format of the M-Bus frame will stay unchanged and it will be still possible to use any M-Bus parser, but the final payload will be shorter.

The VIF DIF filtering can be applied by uncommenting the section below:

```
-- UNCOMMENT BELOW IN CASE OF USING VIF DIF FILTERING
    --api.mbusVifDifFilter("populate",pack.pack("b18",  0x05,         -- 0x05 - number of DIF VIF values
                                      -- 0x03, 0x84, 0x40, 0x06,     -- Energy counter reading - kWh
                                      -- 0x03, 0x84, 0x40, 0x16,     -- Volume counter reading - m3
                                      -- 0x03, 0x85, 0x04, 0x2b,     -- Power [W]
                                      -- 0x03, 0x85, 0x40, 0x3b,     -- Flow [l / h]
                                      -- 0x03, 0x05, 0x83, 0x33))    -- Temperature difference [K]
    -- When applying VIF DIF filter do not forget to uncomment the line 42
    -- Example is taken from device Calec ST 3
```

Each filter consists of the:

- 1st byte indicating length of the filter
- DIF value(s)
- VIF value(s)

Configuration example

The example is taken from the heat meter Engelmann SensoStar U. The goal is to filter following values - Energy, Volume counter reading, Power, Volume flow and Temperature difference. By filtering only for these values we should be able to stay under 51B and the data will be transmitted in one LoRaWAN message.

To do so, we need to configure the VIF DIF filter:

1. Uncomment the line 48 - `--api.mbusVifDifFilter("activate", 0)` to `api.mbusVifDifFilter("activate", 0)`

2. Uncomment the lines 121 to 126 -

```
 api.mbusVifDifFilter("populate",pack.pack("b18",  0x05,                    -- 0x05 - number of DIF VIF values
                                                0x03, 0x84, 0x40, 0x06,     -- Energy counter reading - kWh
                                                0x03, 0x84, 0x40, 0x16,     -- Volume counter reading - m3
                                                0x03, 0x85, 0x04, 0x2b,     -- Power [W]
                                                0x03, 0x85, 0x40, 0x3b,     -- Flow [l / h]
                                                0x03, 0x05, 0x83, 0x33))    -- Temperature difference [K]
```

3. Change the filter to correspond with a VIF DIF values used by the meter -

```
api.mbusVifDifFilter("populate",pack.pack("b16",  0x05,                    -- 0x05 - number of DIF VIF values
                                                0x02, 0x04, 0x06,     -- Energy - Wh
                                                0x02, 0x04, 0x13,     -- Volume counter reading - m3
                                                0x02, 0x14, 0x2b,     -- Power [W]
                                                0x02, 0x4, 0x3b,      -- Volume flow [m3 / h]
                                                0x02, 0x02, 0x61))    -- Temperature difference [K]
```

💡 Do not forget to check the first byte within the VIF DIF filter and if it is indicating the correct length of the filter - **0x02**, **0x04, 0x13**,…

4. Make sure to check the “b..” value. This specifies the number of bytes within the function and has to be corresponding with a number of following bytes. Documentation of this specific function can be found here: https://wiki.acrios.com/en/app_notes/an012

```
api.mbusVifDifFilter("populate",pack.pack("b16", ...
```

Payload example

From MBus Meter:

```
68 2B 2B 68 08 00 72 98 53 85 30 C5
 14 00 0A D1 10 00 00 04 06 00 00 00
00 04 13 00 00 00 00 14 2B 00 00 00
 00 04 3B 00 00 00 00 02 61 14 00 F4
16
```

Parsed data:

```
ID: 30855398
MAN:EFE
MED:COOL_OUTLET
GEN:0

Identification
Product name Engelmann SensoStar 2
Serial number EFE0030855398
Medium Cool outlet
Generation 0

Values
29.08.2023 09:04 Heating energy 0 Wh
29.08.2023 09:04 Volume 0 m³
29.08.2023 09:04 Power Max 0 W
29.08.2023 09:04 Volume flow 0 m³/h
29.08.2023 09:04 Temperature diff 0,
2 °K
```

💡 It is possible to get a list of VIF DIF values by using Alexander Miller parser available at https://www.miller-alex.de/Mbus. It is needed to turn on the “Expert” mode

```
HEADER
      ID: 30855398 MAN:EFE MED:COOL_OUTLET GEN:0 ACC:141 STA:16 SIG:0
RECORDS
DIF:04h (Int32) VIF:78h Data:E6D0D601h FAB number 30855398
DIF:04h (Int32) VIF:06h Data:00000000h Energy 0 Wh
DIF:04h (Int32) VIF:13h Data:00000000h Volume 0 m³
DIF:04h (Int32) VIF:2Bh Data:00000000h Power 0 W
DIF:14h (Int32) VIF:2Bh Data:00000000h Power 0 W Max
DIF:04h (Int32) VIF:3Bh Data:00000000h Volume flow 0 m³/h
DIF:14h (Int32) VIF:3Bh Data:00000000h Volume flow 0 m³/h Max
DIF:02h (Int16) VIF:5Bh Data:1800h Flow temperature 24 °C
DIF:02h (Int16) VIF:5Fh Data:1800h Return temperature 24 °C
DIF:02h (Int16) VIF:61h Data:C9FFh Temperature diff -0.55 °K
DIF:02h (Int16) VIF:23h Data:B900h On time 185000 day
DIF:04h (Int32) VIF:6Dh Data:082AFC28h  28.08.2023 10:08
DIF:44h (Int32) VIF:06h Data:00000000h Energy 0 Wh Storage:1
DIF:44h (Int32) VIF:13h Data:00000000h Volume 0 m³ Storage:1
DIF:42h (Int16) VIF:6Ch Data:0000h  --- Storage:1
DIF:01h (Byte) VIF:FDh VIFE:17h Data:10h Error flags 16
DIF:03h (Int24) VIF:FDh VIFE:0Ch Data:050000h Model 5
DIF:84h (Int32) DIFE:20h VIF:06h Data:00000000h Energy 0 Wh Tariff:2
DIF:C4h (Int32) DIFE:20h VIF:06h Data:00000000h Energy 0 Wh Storage:1 Tariff:2
DIF:84h (Int32) DIFE:30h VIF:06h Data:00000000h Energy 0 Wh Tariff:3
DIF:C4h (Int32) DIFE:30h VIF:06h Data:00000000h Energy 0 Wh Storage:1 Tariff:3
```

Troubleshooting & FAQ

Device is not connecting to the Network Server

- Please check if the inserted keys are correct, you have device configured in OTAA and the AppEUI is not required. In case AppEUI is required, please use the same “0” - 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00.

Device is sending unknown payload

- Please check if the payload is “4E 4F 20 44 41 54 41 20 52 45 43 45 49 56 45 44” - in which cases it translates to “NO DATA RECEIVED” in ASCII.

Device is still sending NO DATA RECEIVED

- Please check if the M-Bus configuration is correct - baudrate and parity and if the electrical connection is done properly. You can also try to change parameter “initialDelay” to a larger value as some meters require up to 6 seconds = 6000 ms. Check that no more than one device is connected without any prior changes to configuration.

Device is not connecting to the GUI

- Please make sure to use the Chromium based browser, such as Google Chrome. Make also sure that the serial line is not opened in any other serial line monitor.

Device has connected, LUA script was uploaded but it is not possible to connect anymore

- Make sure the battery has been disconnected for a longer period of time to discharge the capacitor or alternatively short the battery pins on the PCB. Connect two metal pins in the battery connector on the PCB with something conductive (tip of screw driver, paper clip, tip of a pen…). The device can connect only when in bootloader or when it is sleeping. If the device is in the application LUA script and currently running, it will not connect.

Where do I configure the LUA script?

- www.gui.acrios.com. Please make sure to use Chromium based browser.

Where can I see a data or serial line log?

- You can check any serial line monitor such as PuTTy or Termite. Please make sure the serial line monitor configuration is - baud rate 115 200, data bits 8, stop bits 1, parity none.

I am getting a one-byte answer. What does it mean?

- The 1-byte answer usually means there is a collision on the M-Bus line. This typically occurs when more than one M-Bus meter is connected and the M-Bus converter is sending a broadcast query on address 254. The exception might be a “0xE5” which is a confirmation from the meter according to M-Bus standard.

I applied VIF DIF filter but I am getting last error traceback in the serial line - what could be wrong?

- Please check the function `api.mbusVifDifFilter` and that the specified number of bytes corresponds with actual number of bytes within the function. The byte counter includes the first byte representing number of VIF DIF values.

I want to apply VIF DIF filter but the documentation doesn’t say anything about M-Bus frames. What should I do?

- It is possible to get a list of VIF DIF values by using Alexander Miller parser available at https://www.miller-alex.de/Mbus. It is needed to turn on the “Expert” mode.

I want to have a possibility to remotely configure hardware and the LUA script. How can I do it?

- You can either take a look at our documentation that is available at wiki.acrios.com and modify the LUA script by yourself or you can contact our support at support@acrios.com which should point you towards the right solution.

I did not find my question here.

- Please check wiki.acrios.com or contact us at support@acrios.com